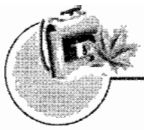

第1章

搭建工作环境

“工欲善其事，必先利其器”。在我们开始写操作系统代码之前，花些时间来学习工具的使用其实是非常必要的。从过程上看，操作系统的开发与普通应用程序的开发并没有太大的区别。正像开发应用程序那样，要开始一个操作系统的开发，首先也必须要解决开发环境搭建的问题。

搭建开发环境归根结底是要解决四个问题：

- 在什么样的系统环境下开发？
- 使用什么样的编辑工具？
- 怎样编译程序？
- 程序如何运行？

为了能够帮助读者学习本书的内容，在本章中我们将会围绕上述四个问题展开深入的讨论，以帮助读者顺利地编写出属于自己的嵌入式操作系统。

1.1 选择合适的开发环境

每一个计算机爱好者都有自己心仪的操作系统。有些开源软件和嵌入式技术的爱好者也许偏好 Linux，一些追求时尚和个性的朋友或许钟爱 Mac OS，但对于绝大多数读者来说，Windows 操作系统应该还是最熟悉的。

本来使用什么样的操作系统，不应该成为限制开发的理由。但这里需要说明的是，本书的所有示例代码都是在一个叫做 Gentoo 的 Linux 发行版中开发的。书中所使用的绝大多数工具在 Linux 下都有原生的支持。因此如果有可能，还是建议大家使用 Linux 操作系统来学习和编写本书的示例。

然而，要求那些不熟悉 Linux 操作系统的开发者在短时间内学会使用 Linux 也并不现实。好在 Windows 操作系统在兼容性和应用程序多样性等方面是无人能敌的，于是在 Windows 下，我们同样也找到了一套解决方案来编译和运行本书的代码，它就是 Cygwin。

Cygwin 是一个在 Windows 平台上运行的 UNIX 模拟环境，是 Cygnus Solutions 公司开发的自由软件。它对于学习 UNIX/Linux 操作环境、从 UNIX 到 Windows 的应用程序移植，或者进行某些特殊的开发工作（尤其是使用 gnu 工具集在 Windows 上进行嵌入式系统开发）都非常有用。

下面我们就来介绍一下 Cygwin 环境的安装和使用方法。如果您是 Linux 用户，就可以跳过这部分，继续阅读下一小节的内容。

1.1.1 准备 Cygwin 开发环境

首先，我们需要从 www.Cygwin.org/Cygwin/setup.exe 处下载最新版的 Cygwin 安装程序。

接下来运行这个安装程序，如图 1-1 所示。

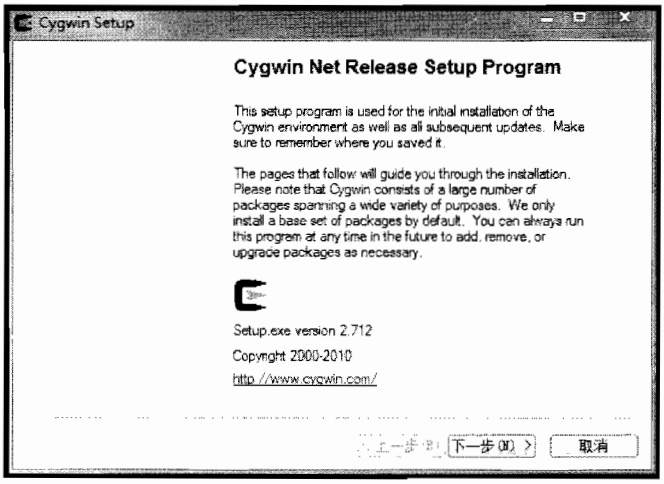

图 1-1 Cygwin 的安装

在图 1-1 的对话框中单击“下一步”按钮，进入安装类型选择页面，如图 1-2 所示。

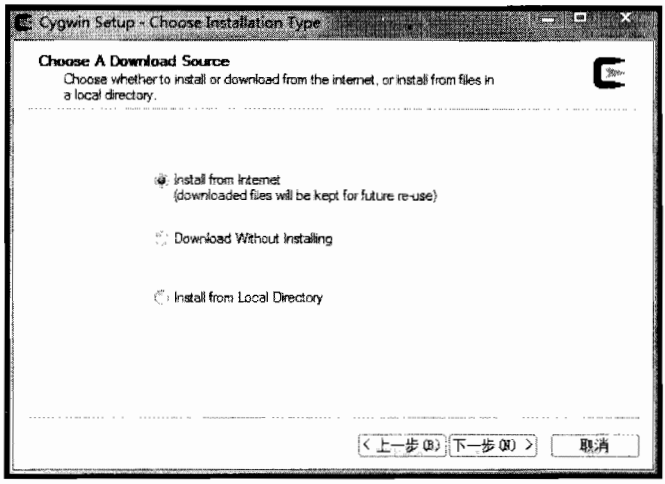

图 1-2　Cygwin 安装类型选择页面

此处如果没有特殊的要求，可以采用默认的“Install from Internet”单选项，通过网络安装 Cygwin 系统。当然，这要求读者的电脑能够上网才可以。

继续单击“下一步”按钮，进入安装路径选择页面，如图 1-3 所示。在这里，读者可以根据自己系统的情况选择某一安装路径。

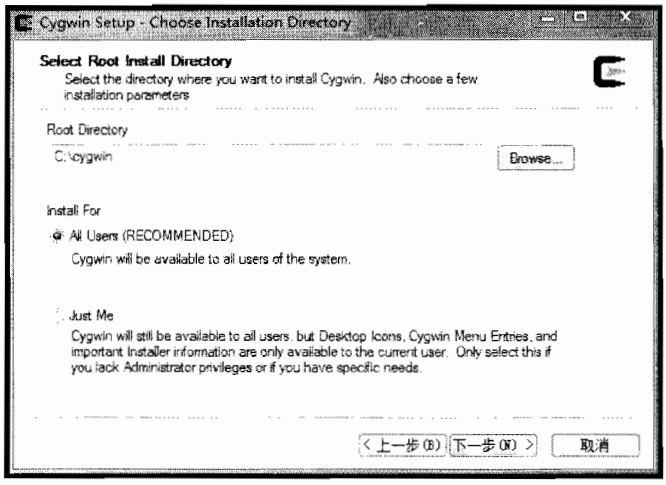

图 1-3　Cygwin 安装目录选择页面

再次单击“下一步”按钮，程序进入软件包下载目录选择页面。之前我

们已经选择了通过网络安装的方法来进行安装，那么在这个页面中，我们就需要指定一个本地文件夹，保存从网上自动下载的各种应用软件包，如图1-4所示。

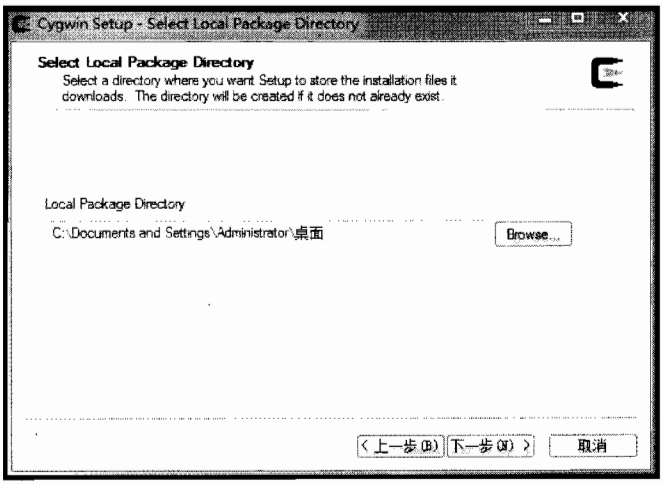

图 1-4　Cygwin 软件包下载目录选择页面

之后程序将打开连接方式选择页面，如果没有特殊要求，这一步选中默认的“Direct Connection”单选项即可，如图1-5所示。

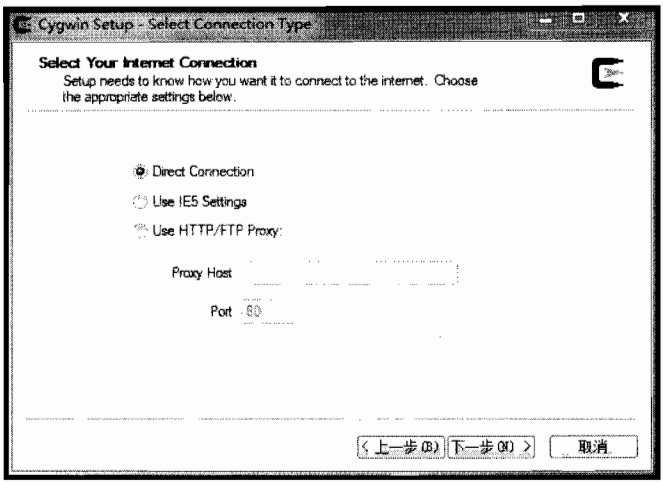

图 1-5　Cygwin 连接方式选择页面

单击“下一步”按钮，进入下载站点选择页面。通常我们会选择http://www.cygwin.cn/作为下载源，如图1-6所示，对于国内的用户来说，这个站点的连接速度最快。

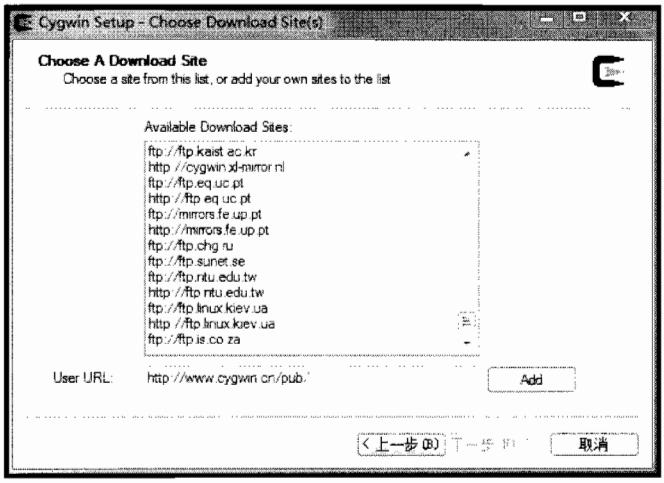

图 1-6　Cygwin 下载站点选择页面

如果是首次运行 Cygwin 安装程序，则需要在“User URL”文本框中添加“http://www.cygwin.cn/pub/”这一地址，单击“Add”按钮。在其他情况下，只需在“Available download Sites:”列表框中选择“http://www.cygwin.cn”站点即可。下载源选择完成后，单击“下一步”按钮，进入软件包选择页面，如图 1-7 所示。

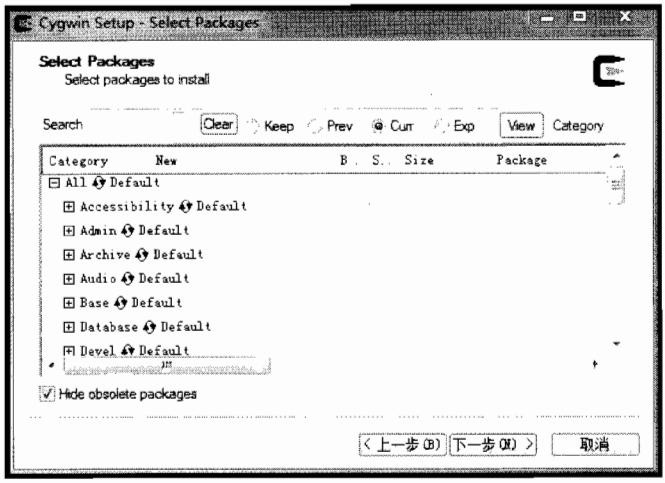

图 1-7　Cygwin 软件包选择页面

在该页面中，我们可以选择需要安装的软件包。由于 Cygwin 默认安装的软件包不能完全满足我们自己的操作系统开发，因此需要至少选择两个额外的软件包进行安装。单击“Devel”类前面的“+”，将该分类展开，从中选择“make”和“gcc4”两组软件包，而其他软件包按照默认方式处理即

可，完成操作后，单击“下一步”按钮，程序进入安装过程，如图 1-8 所示，显示安装进度。

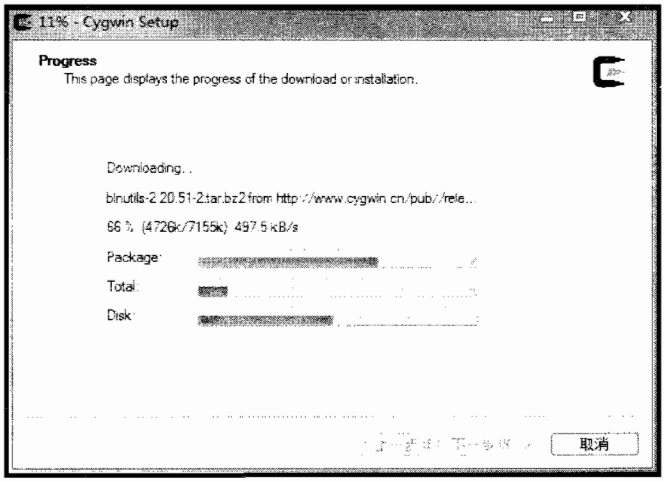

图 1-8　Cygwin 安装过程

片刻之后，Cygwin 的安装就完成了，如图 1-9 所示。

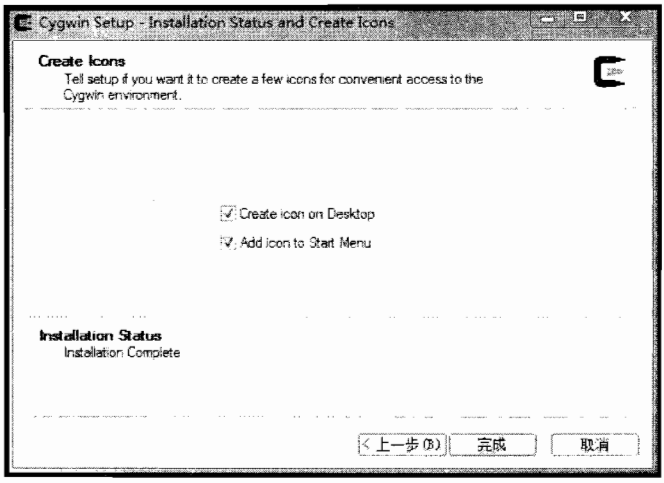

图 1-9　Cygwin 完成安装

单击“完成”按钮，退出 Cygwin 的安装过程。

Cygwin 的使用方法非常简单，只需要单击桌面上的 Cygwin 图标，就会弹出一个类似于 Windows 命令行的页面，如图 1-10 所示。

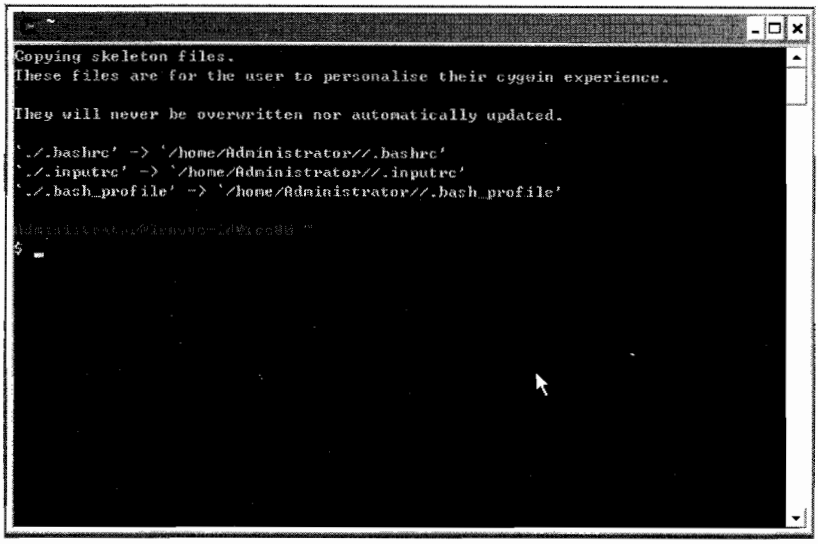

图 1-10　Cygwin Shell 页面

简单地讲，如图 1-10 所示的操作页面实际上是在 Windows 操作系统中模拟出来的类 UNIX 系统中的 Shell 页面，我们可以输入一些 UNIX 下常用的命令，如 ls、cd 等，这些命令的使用方法和运行结果与 UNIX 系统是一致的。对于那些使用 Windows 操作系统的朋友们来说，属于我们自己的操作系统也将在这样一个环境中诞生！

1.1.2　使用 Linux 开发环境

如果读者朋友已经准备使用 Linux 来开发属于自己的操作系统，又没有把握在电脑中直接安装某个 Linux 发行版，那么一个折中的办法就是使用虚拟机，就好像我们又多了一台电脑一样。

目前流行的虚拟机软件五花八门，各有优劣。使用什么样的虚拟机安装 Linux 系统，其实并没有太大的差别。在这些虚拟机中安装 Linux 操作系统的大致步骤如下。

（1）选择读者喜欢的 Linux 发行版，主流的如 ubuntu、redhat，个性的如 gentoo、archLinux，极端的如 LFS。

（2）从各发行版的网站上下载安装光盘镜像，这些光盘镜像有的可能很小，需要通过网络安装的方式进行，有的也许很大，直接一张光盘就可以完

成一个基本系统的安装。

（3）将下载下来的光盘镜像插入到虚拟机中，根据各 Linux 发行版的相关文档或软件提示进行 Linux 的安装。

出于篇幅的原因，本书不对虚拟机下安装 Linux 发行版的详细步骤进行介绍。因为如果介绍详细步骤，一连串的关于 Linux 原理、基本操作方法等问题都将被牵扯出来，本书很有可能就会变成一本 Linux 入门级书籍。为避免喧宾夺主，我们建议只了解 Windows 操作系统的朋友使用 Cygwin 进行开发。对于那些了解 Linux 的朋友们来说，对于 Linux 如何安装自然会非常清楚，因此也无须多言。

1.2 开发工具的使用

自己动手写操作系统的第二个重要工作是选择合适的编译器和编辑器。

1.2.1 编译器的选择和安装

由于我们的操作系统需要运行在 ARM 体系结构中。因此，在琳琅满目的 ARM 编译器中选择出一款，用来编译操作系统源程序，就成为了必须先要解决的问题。于是我们选择了 GCC。

GCC 的大名，相信读者应该都会如雷贯耳。简单地说，GCC 就是由 GNU 基金会组织开发的一套开源的编译器。根据相关许可，我们可以自由地下载和使用这套编译器，可以随意地修改和重新发布它的源代码，整个过程不需要付一分钱或承担任何责任。这就是我们选择 GCC 作为开发工具的一个最主要的原因。

GCC 的免费获得和使用并不等于质量低劣或功能欠缺。它的功能不逊于任何一款商用 ARM 编译器。这也是我们选择 GCC 作为操作系统开发工具的第二个原因。

接下来我们就来介绍一下 GCC 编译器的安装过程。

GCC 是源码开放的，所以理论上，我们完全可以去网站上下载 GCC 源代码，然后在本地进行编译，生成一个能够编译 ARM 体系结构下程序

的工具。

这个过程听起来似乎轻而易举，但做起来却没有那么容易。况且也没有太大的必要自己生成编译器。因此，本书不会对这一部分内容进行介绍。如果读者确实对自己构建编译器很感兴趣，可以去“www.leeos.org”上翻阅相关文档。

其实，这个问题远远没有那么复杂。编译器也只不过是一套软件而已，别人编译好的，拿到我们的系统中一样可以使用。

通常我们会看到网上有许多名为“arm-Linux-gcc”或“arm-elf-gcc”的编译工具。使用这些工具来编译属于我们自己的操作系统，基本上都是可以的。

读者也许会提出这样的疑问，为什么这些编译器的名字有些与众不同呢？其实，这些编译器的命名方法有些约定俗成的规矩，例如，开头的“arm”关键字表示该编译器将生成 ARM 体系结构机器码，而中间的“Linux”或“elf”则表示应用的目标平台。一般说来，“Linux”关键字意味着该编译器在编译时针对 Linux 系统做了特定的优化，生成的代码将更好地运行在 Linux 系统之中，而“elf”关键字则更适合生成通常的可执行程序代码。对于我们自己的操作系统来说，使用“arm-elf-gcc”这套工具显然更合适一些。与传统的 PC 程序开发不同，嵌入式的开发过程通常都是在 PC 中对源代码进行编译，再将编译生成的可执行程序放到嵌入式平台中去运行，因此有人把这种编译过程称作交叉编译或跨平台编译，而将一整套用于交叉编译的工具称为交叉编译工具链。

为了方便读者对本书代码进行实践，我们制作好了一套专门用于这套操作系统的编译器，大家可以去“www.leeos.org”网站上下载使用。

需要注意的是，这样的一套编译工具只能用来编译我们自己写的操作系统，不能保证编译其他 ARM 应用程序的正确性，包括 Linux 下的程序和内核。这是因为在制作这套工具的过程中，为了能让编译器尽可能小一些，我们没有添加任何标准 C 函数库和其他一些 GCC 扩展工具，对于我们自己的操作系统来说，这样做并无问题。也就是说，通常情况下，一个操作系统的核心代码并不需要第三方函数库，但这个假设对于绝大多数应用程序来说却并不成立。

当下载到某一个版本的交叉编译工具之后，第一步要做的就是将其解压。

以我们所提供的交叉编译工具链为例，Windows 用户可以运行如下命令。

命令 1-1

```
tar zxvf leeos_tools_for_Cygwin.tar.gz
```

当然这需要首先将下载的编译工具复制到某一目录中，如果读者将 Cygwin 安装到了 C 盘的话，这个目录可以是“C:\Cygwin\usr”，然后打开 Cygwin 命令行，运行下面的命令。

命令 1-2

```
cd /usr
```

这样，我们就可以使用命令 1-1，将安装工具解压到“usr”目录下。此时虽然解压过程已经完成，但并不表示编译器已经可以使用，我们还需运行如下命令才能将编译器彻底地安装到系统中。

命令 1-3

```
echo "PATH=\$PATH:/usr/leeos_tools_for_Cygwin/bin" >>/etc/profile
```

对于 Linux 用户，步骤基本是一样的。但需要注意的是这些命令需要以超级用户的身份去运行。使用命令 1-4 可以进行超级用户的切换。

命令 1-4

```
su
```

使用 su 命令后，终端会提示输入超级用户密码。用户切换完成后，就可以使用 cp 命令将下载的编译工具复制到“/usr”目录中。

命令 1-5

```
cp /where/your/compiler/locate/leeos_tools_for_Linux.tar.gz /usr
```

接下来运行命令 1-2，然后是命令 1-1 和命令 1-3。注意，要将“leeos_tools_for_Cygwin”改成“leeos_tools_for_Linux”。重启系统后，交叉编译工具就可以使用了。

1.2.2 编辑器的选择和使用

编译器已经有了，我们需要使用什么工具来编写操作系统代码呢？

其实，只要自己觉得合适，无论是使用简单的（诸如记事本之类的）工

具还是复杂的（诸如进行程序开发的专用 IDE），都不会有问题，每个人心中应该都有自己最理想的编辑工具。在程序的开发过程中，我们只需要打开一款编辑器，将程序源代码敲进去，然后将这些内容保存成源文件，那么剩下的工作就是使用编译器对这些代码进行编译。

接下来，让我们通过 Windows 下的“记事本”这一工具来演示一下在 Cygwin 环境中编写代码的一般方法，其他代码编辑器的使用与该方法类似。

首先请打开文本编辑器，向里面输入一个空的 main 函数，如图 1-11 所示。

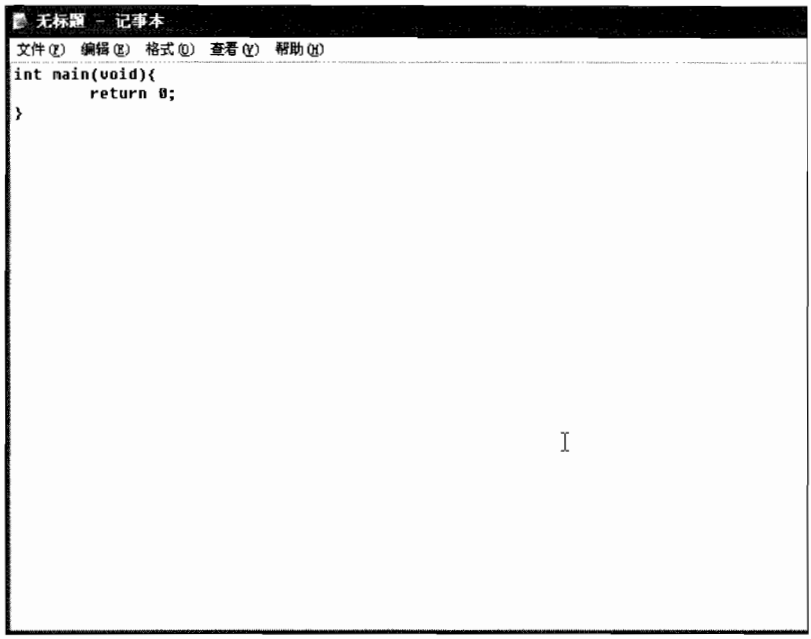

图 1-11　使用编辑器编写代码

接下来我们尝试对这段代码进行编译。将这些内容保存成文件，取名为“test.c”。在 Cygwin 的安装目录下，有一个“home”文件夹，进入这个文件夹，可以看到以用户名命名的一个文件夹，该文件夹代表 Cygwin 用户的家目录。所有个人私有的文件和数据存放到这个目录中是比较合适的。

例如，某个人将 Cygwin 安装到了 C 盘根目录，同时他又是以 Administrator 用户登录的话，那么这个文件夹就应该是 C:\Cygwin\home\Administrator。

保存完成后，运行 Cygwin，并在弹出的命令行中输入“ls”这条命令，

可以看到 test.c 文件出现在该目录下，如图 1-12 所示。

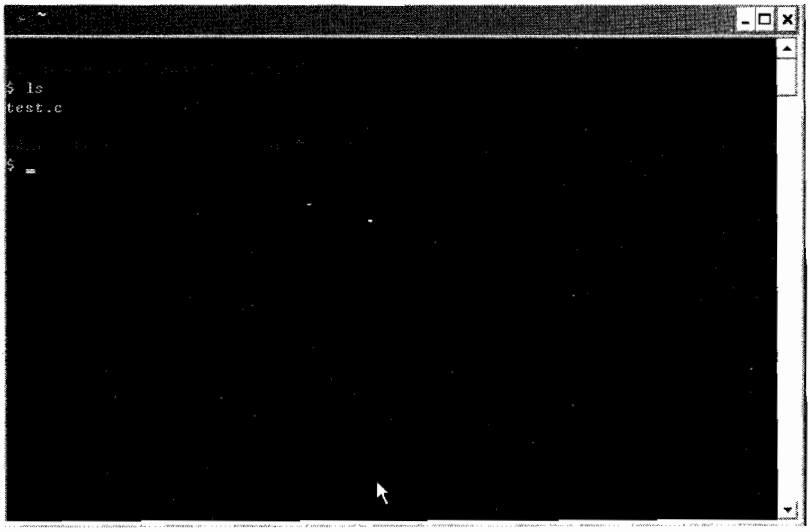

图 1-12　在 Cygwin 命令行运行“ls”命令

紧接着就在这个命令行下，运行编译工具，将“test.c”文件编译成可执行程序，命令如下。

命令 1-6

```
arm-elf-gcc -nostdlib test.c
```

最终在该目录下，会出现一个名为“a.out”的文件，这个文件正是由 GCC 编译 test.c 文件后生成的。这就表示我们已经使用编辑器编写了代码，并用编译器成功编译了第一个程序。

对 GCC 比较熟悉的读者可能会觉得命令 1-6 这种编译方法比较奇怪。如果您使用的是我们提供的专用编译器，那么“-nostdlib”参数是必需的。它表示编译时不去链接标准函数库。相关参数的具体含义，我们会在接下来的章节中详细阐述。

如果是 Linux 用户，那么程序的编写和编译过程会更简单些。因为在 Linux 下，存在有一些基于命令行的文本编辑器，如 vi。使用这样的一些编辑器，程序的编写、代码的保存和编译等过程都可以在命令行下实现，这样，开发效率会较高。

如果一些 Linux 初学者觉得 vi 较难掌握，也可以尝试使用 nano。nano 也是一个在命令行下就能运行的文本编辑器，它的用法像 Windows 下的“记

事本”一样简单，图 1-13 是一个在 Linux 下运行 nano 的页面。

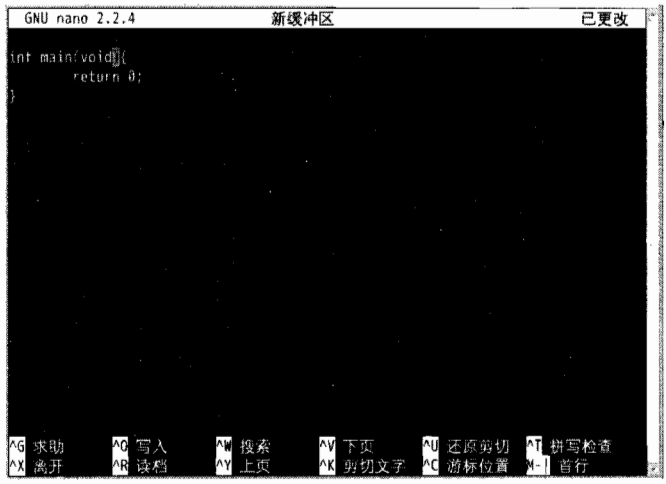

图 1-13　nano 运行页面

在确保系统中安装了 nano 工具的前提下，想要使用 nano，只需要首先通过 cd 命令切换到某一路径下，然后运行下面的命令。

命令 1-7

```
nano test.c
```

这样就可以打开一个 test.c 文件，并进入文本编辑的页面中了。在写入适当的内容之后，按下“Ctrl+O”组合键可以保存文件，按下“Ctrl+X”组合键可以退出页面。

如果有些朋友还是觉得使用图形化的工具更加方便，那么，在 Linux 下，也有很多图形化的文本编辑器可供选择，如 gedit、gvim、mousepad，等等。这些工具的使用方法与 Windows 下的编辑工具完全一致。最终我们还是要通过命令行来编译源程序。

1.3　虚拟硬件的安装和使用

现在虽然我们已经编译生成了一个应用程序，但却没有办法运行它。其中的道理很简单，我们使用 ARM 编译器交叉编译生成的可执行程序，自然

是只有在 ARM 硬件环境下才能够使用。

一个最直接的方法，就是使用 ARM 开发平台。市面上基于 ARM 的开发板多如牛毛，无论选择哪一款，都可以帮助我们将操作系统运行起来。但很显然，这种方法并不具备可操作性。其中的一个原因是这样将会无端地增加我们的学习成本。另一个原因是，由于各种开发板的芯片选型、电路设计都不尽相同，使用这些开发平台进行开发，我们将会陷入无尽的硬件细节当中，将不能够从宏观的角度去理解操作系统的原理。

为了解决这个问题，我们需要借助一种特殊的软件来运行编译器生成的 ARM 程序，这就是虚拟机。

虚拟机是运行在 PC 中的一种软件，能够模拟出 ARM 硬件环境。这样一来，我们既不必在操作系统的编写和学习过程中多花一分钱，又能拥有一个统一的硬件平台，不至于牵扯过多的硬件细节，迷失到森林之中。

有很多免费的 ARM 虚拟机可供选择，SkyEye 正是其中之一。

SkyEye 是一个开源软件项目，其目标是在通用的 Linux 和 Windows 平台上实现一个纯软件集成开发环境，模拟常见的嵌入式计算机系统。我们可在 SkyEye 上运行μClinux 以及 μC/OS-II、Linux 等多种嵌入式操作系统和各种系统软件，并可对它们进行源码级的分析和测试。SkyEye 是一个指令级模拟器，可以模拟多种嵌入式开发板，可支持多种 CPU 指令集，并支持网络、Flash 等大量硬件。在 SkyEye 上运行的操作系统意识不到它是在一个虚拟的环境中运行的。值得一提的是，SkyEye 项目是由清华大学的博士后陈先生发起的，是属于我们中国人的优秀软件项目。

与 SkyEye 类似的虚拟机还有 QEMU。从某种程度上说，QEMU 较之于 SkyEye 更加优秀。但是本书最终选择了 SkyEye 而放弃了 QEMU，主要有以下两个原因：

第一，SkyEye 易于配置，方便使用。我们可以通过修改配置文件，轻松地搭建出具有任何硬件特性的虚拟平台。

第二，SkyEye 对时下比较流行的几款 ARM 芯片支持良好，如 SAMSUNG 的 s3c 系列及 ATMEL 的 AT91 系列等。拥有 ARM 知识基础的读者对这两个系列的芯片应该都有所了解。以这些常用芯片为基础进行操作系统的开发，将有利于为那些只是略懂硬件的读者拨开硬件迷雾，更清晰地看清楚操作系统的全貌。

ARM 虚拟机既已选定，接着我们就来了解一下 SkyEye 的安装与使用。

1.3.1　SkyEye 的安装

SkyEye 的安装不需要多说，只需要从官方网站上下载 SkyEye 源代码，然后复制到本地文件夹中，并在命令行中输入相应命令即可。如果以 1.2.6_rc1 版本为例，那么安装命令如下。

命令 1-8

```
tar jxvf skyeye-1.2.6_rc1.tar.bz2
cd skyeye-1.2.6_rc1
./configure
make
make install
```

如果是 Linux 用户，在运行 make install 命令前可能需要暂时切换到超级用户下，这可以通过命令 1-4 实现。

由于每个人的系统环境可能会略有差别，我们提供的这种安装方法不能保证一定会成功。即使如此，读者也不用担心，我们已经预先编译好了 SkyEye 可执行程序，读者也可以去“www.leeos.org”下载。

对于 Windows 用户，可以将下载下来的“skyeye.exe”文件复制到 Cygwin 安装目录中的“usr/bin”文件夹下。而对于 Linux 用户，可以下载 skyeye 这个文件并复制到“/usr/bin”目录中。这样，ARM 虚拟机就安装完成了。

1.3.2　SkyEye 的使用

想要使用 SkyEye，就必须要了解它的配置方法。skyeye.conf 文件是 SkyEye 的默认配置文件。通过编写合适的 skyeye.conf 文件，我们可以配置出任何 SkyEye 支持的硬件环境。

代码 1-9 是一个比较典型的 skyeye.conf 配置文件的范例，针对的是 s3c2410x 这款芯片。

代码 1-9

```
cpu:  arm920t
```

```
    mach: s3c2410x
    mem_bank:map=M, type=RW, addr=0x30000000, size=0x00800000,
file=./leeos.bin,boot=yes
    mem_bank: map=I, type=RW, addr=0x48000000, size=0x20000000
```

第一行中的关键字 cpu 记录的是芯片系列，第二行中的 mach 关键字记录的是芯片的具体型号。

接下来的 mem_bank 关键字描述的是芯片内存空间特性。其中，map=M 代表该段内存空间是一段内存，而如果是 map=I，则代表该内存空间对应的是外设端口。type=RW 表示的则是该内存空间具备可读写属性。之后的 addr=0x30000000 和 size=0x00800000 分别代表内存空间的起始地址和大小，这里，我们将虚拟开发平台配置为拥有 8M 内存并开始于 0x30000000 处。后面的 file=./leeos.bin 表示的是预先要被加载到这段内存空间的映像文件。而 boot=yes 则表示默认从此处启动。有了这两个属性，在操作系统调试或运行时就不必关心程序的引导，而可以直截了当地去运行了。需要注意的是，mem_back 关键字之后的内容需要出现在一行里，由于版面的原因，这里被分成了两行。

关于 SkyEye 配置文件的写法，我们已经了解，下面介绍它的运行方法。

首先我们需要将代码 1-9 保存成文件，名为“skyeye.conf”，并放到某一个文件夹下。其次，需要将编译完成的二进制映像文件放到同一个目录下，这个二进制文件的名字应该与 skyeye.conf 文件中记录的文件名相一致。最后，只需要开启一个命令行，输入 skyeye 命令，程序就可以运行了。整个过程如图 1-14 所示。

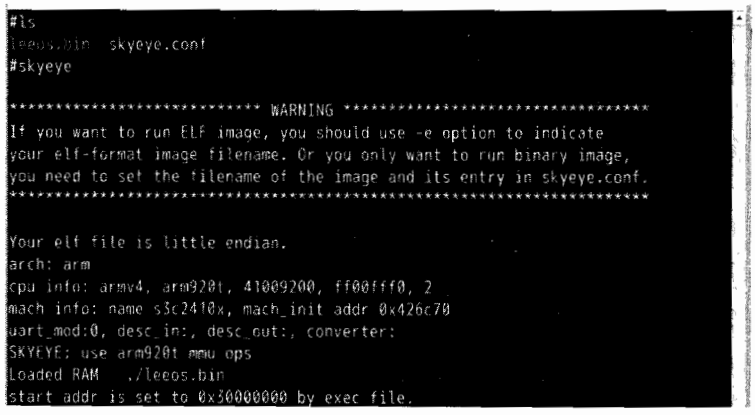

图 1-14　SkyEye 的使用方法

最后还要强调，本书所有的示例代码都是运行在由 SkyEye 模拟的 s3c2410 这一硬件平台，原因就像前面介绍的，s3c2410 在国内比较流行，了解的人较多。同时，SkyEye 对这一平台的支持也非常成熟且资源丰富，这些对操作系统的开发都非常有利。如果读者对其他平台有所了解，将本书的例子代码移植过去也不是什么难事。

1.4　总结

本章我们主要介绍了与开发嵌入式操作系统相关的一系列工具及其使用方法。这些工具方法都具备免费获取、使用简单等特点。它们有可能不是最优秀的，但恰恰都是最适合的。

其实一件作品的完成，关键并不在于工具，而是在于使用工具的人。美国人可以在计算机技术尚不发达的情况下就让航天飞机上天，中国人亦可在经济条件极度困难的情况下爆破原子弹。倘若胸怀愚公之志，纵使手挑肩担，也能将大山夷为平地。

不过这个问题似乎说远了，其实自己一步一步地去写一个操作系统，远没有那么大的难度。不信？那就接着读下去吧！